

Study on the undirected graph function for ship scantling calculation

Case Study of Finding All Possible Cycle by Modified DFS on the Ship Structural Member

Triyan Indrawan^{a*}, Hardika Raditya A.^b, Sukron Makmun^c, and Topan Firmandha^d

Research and Development Division, PT Biro Klasifikasi Indonesia (Persero), Jakarta, Indonesia

a. triyan@bki.co.id, b. hardika.ra@bki.co.id, c. sukron@bki.co.id, d. topan@bki.co.id

**corresponding author*

Keywords: Ship scantling, DFS, undirected graph, open loop, closed loop.

Abstract: Ship's structural strength is one of technical aspect which strongly affects the safety of ship. In general, ship's construction structure will experience load that coming from internal effect from ships deadweight and external effect (environment). The amount of the load at a part of ship's construction will affect the dimension and thickness of the corresponding structure, thus appropriate scantling calculation will be a very important factor in anticipating the loading condition which will occur in the ship. However, calculating actual stress is a fairly complex process, hence manual process of calculating plate thickness for whole ship will consume too many hours. To solve that problem, a scantling calculation application is developed using computer program. The application is created by taking into account the geometrical form of ship's cross section, since the calculation is variated by that form. To reach that purpose, geometrical recognition algorithm for cross sectional structure is developed in order to detect the structural form. However, computer calculation possesses a significant challenge namely identification of closed loop and identification of loop section. In a manual calculation, the closed loop or open loop section of the ship construction can be identified easily by human sight. Computer, on the other hand, can only detect an object consists of sets of lines and node instead of the actual form of structural cross section. Hence, it is unable to fully recognized closed and open loop section. The solution that can be used to solve this problem is by exploring all the lines and node available and try to recognize closed loop with DFS Method. This research is focused on developing enhanced DFS algorithm to increase accuracy in recognition and classifying them into closed or open loop to reach the target of accurately identify all the closed loop.

1. Introduction

In general, a construction consist of various structural parts which combined together to performed a robust and safe construction design against any form of load. In the same principle, ship's construction also formed by many structural parts which supports each other structural members. Every structural components need to be verified regarding its strength and safety factor in accordance

with required safety and construction standards. Therefore, many calculation approach to assess ship structural strength have been developed. The most convenient way for this purpose is by using prescriptive formula, with consideration of the assumptions made on those calculation are nearly close to the actual condition [1].

However, with so much variations of ship's hull body shape, prescriptive formula approach will tend to be inaccurate and – even more, unrealistic. One of good example which describe this situation is the assumption of Bending and Shear stress at ship's cross section [2]. This prescriptive calculation treats all variant of hull body shape with the same assumptions in regards to its bending and shear stress. This kind of approach is pretty much contradictory with prismatic block theory, in which it states that the magnitude of bending moment will be proportional to geometrical shape and cross sections. Moreover, with the addition of torsional moment for ships with large opening on its deck, e.g. container ship, the calculation would seem more and more complex with various scheme of open and closed cross section.

The requirement for ship structural assessment with regards to safety including plate thickness, stiffeners and girder dimensional size, etc. is also known as structural scantling. In the Rules of Biro Klasifikasi Indonesia (BKI) as a class society, all of minimum scantling requirement needs to be fulfilled in order to ensure the design construction's safety [3]. Those scantling calculation magnitude will be affected by the magnitude of static load, dynamic load and stress on the ship cross section.

Along with the rapid development of digital tools in this modern era, various high-end calculation software has been developed. This, however, also brings another challenge with regard to the inability of computer to actually look at the drawing design. Hence, the solution of this problem leads to the development of calculation algorithm to solve various possible hull body shape and design constructions. This kind of algorithm needs to be continuously developed in order to accommodate analytical assessment on every possible hull design. To solve the above-mentioned problem, those hull geometrical cross section can be treated as a group of closed cycle loop and open cycle loop.

Therefore, this research concentrates mostly on development of suitable calculation algorithm using certain method known as DFS (Depth First Search). Using this approach, the computer treats ship's hull design as a group of node and lines which forms a segment. Then, each segment is explored and recognized by the computer as closed cycle loop segment, open cycle segment or combination of both. Finally, a suggestion is proposed in this paper, which is one of procedure of those cycle loop recognition including its proof of accuracy.

2. Graphical Theory Definition

The classical problem of efficiently listing all the simple cycles in a graph has been studied since the early 70s. For a directed graph with n -vertices and m -edges, containing u -cycles, the most efficient solution was presented by Johnson [4] and takes $O = ((u+1)(n+m))$ time. This solution is not optimal for undirected graphs; nevertheless, no theoretical improvements have been proposed in the past decades [5].

For undirected graph, G , it will be consisted of a group of node (vertex) and edge sets, which from now will be known as function of (V,E) in which V is a group of coordinate node and E is part of matrix $V \times V$ sets from its edge. This means each of the edge always have a node. In the graphical geometry which also have a direction, each edge will have a direction on every nodes, hence every edge will consist of tail node and head edge. For a geometry with no direction every node will not have any direction.

Some graphical lines can be formed with limited sequence of each passed node (V_1, V_2, \dots, V_k) , if each note will be represents as i , the magnitude of i is between 1 to $k-1$ the each passed node hence can also be written as $(V_i, V_{i+1})E$. Then all the neighbouring node must have an edge to continue

forming a graphical line, hence there it will formed a continuous sequence from V1 to Vk. In the path to form a graphical lines/geometry, the sequence will never pass a same node, and number of passing edge will be the total length for those graphical lines.

2.1. Definition

- Path is a sequence of node which is connected by its edge.
- Simple path is a traveling sequence of node without any repetition from the same node (made an open loop)
- Cycle is a path which has the same start and end nodes.
- Simple cycle is a cycle without any repetitive edge or nodes

3. Depth First Search (DFS)

DFS algorithm works to search a graphical loop or binary data structural tree. This algorithm will be started from root edge of binary tree and walks along some certain path, hence will produce certain backtracks on edge node which can reveal unexplored path. This algorithm will be repeated until all possible path have been explored (as described on the fig. 1.) [6].

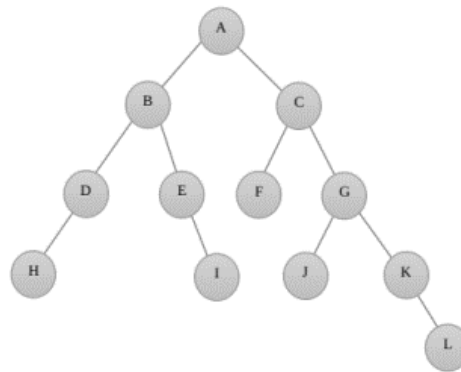


Figure 1: Binary tree.

On the Fig. 1, according to DFS, the sequence formed on this binary tree is A,B,D,H,E,I,C,F,G,J,K,L. Generally, DFS application can be done recursively. With using root as a starting input, in Fig. 1 case is on the vertex A, hence it will be continued with vertex B while marking each passed vertex. Where there is a branch (more than one path available), then this algorithm will choose one of them while marking the explored one, until all of the vertex available is explored.

To simplify DFS implementation, we can use a main component known as “representation graph”, which is a graph that shows cross path that affect traveling path of DFS algorithm.

3.1. Representation Graph

Representation graphic is a simplified solution which consist of two way approach such as adjacency list and adjacency matrix. Both of those things are very important to understand for any programmer that has any interest in developing DFS. The description of both approach can be described as follows:

3.1.1. Adjacency List

This list includes any list of cross path coordinates from node a and forming an edge with nodes (a,b) which connected to node a. This adjacency list then can be saved into an array. See Fig. 2. For detail illustration.

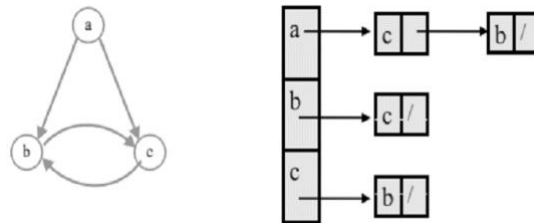


Figure 2: Adjacency list.

3.1.2. Adjacency Matrix

Adjacency matrix is a set of matrix $[V] \times [V]$ and its element can be represented as an (i,j) , the magnitude of each element is 1 when each nodes (v_i,v_j) are forming an edge (E). Adjacency matrix will need large space (constant factor can be negligible because input matrix only in several bits). Detail illustration on Fig. 3.



Figure 3: Adjacency matrix.

3.2. Edge Classification

Several method to classify DFS data binary tree are listed as follow:

1. Tree edge – listing crossed edge found in a edge travelling search.
2. Back edge – is an edge (u,v) which start from the certain node u into the previous starting node at DFS tree
3. Forward edge – is an edge (u,v) that start from the node u into the other unexplored node at DFS tree
4. Cross Edge – other edge outside those description above

3.3. Back Edge

The main important thing in a searching process or observation process where the graphic lines only have a magnitude but did not have any direction is finding back edge. When the first attempt for searching process is arrived on the same node twice, and the other search process found the same node from the unexplored path (from other edge (u,v)), this can means that the search have found two access which is u to v by following the edge and the path which comes from the first searching attempt(back edge). In short, when those above process happened the computer might possibly found a cycle. Hence the minimum requirement to form a cycle is when the cycle is consist of three edges.

3.4.Procedure to Find Cycle

As described before, where searching process from back edge enable computer to have possibly found a cycle. Therefore, any potential edge which can possibly formed a cycle will be saved on variable then in further process another search on the saved edge will be carried out with the same DFS method.

In a nutshell, DFS concept consist of a recursive attempt on searching and exploring all the nodes including each of adjacency cross path until all the nodes are already explored, which means all possible cycle have already been found.

Cycle recognizing process on undirected graph using DFS need more complex effort, such as DFS has to be performed on each node and monitored all the founded path when the search process are going back into the original nodes.

Therefore the detail step to carry on those above process can be listed as follow:

1. Preparation
 2. Recognizing the cycle
 3. Result
1. Preparation stage will carry on these below steps:
- a) Creating adjacency list in order to simplify next algorithm process
 - b) Carry on initial DFS at undirected model and monitoring back edge result. The variable which need to be saved is the present and ongoing node at back edge.

Exploration process using DFS while monitoring back edge result can only detect possibility of cycle presence on the searched part, while the sequence of the node that forming those very own cycle cannot be found. See Fig.4. flow chart for detecting back edge.

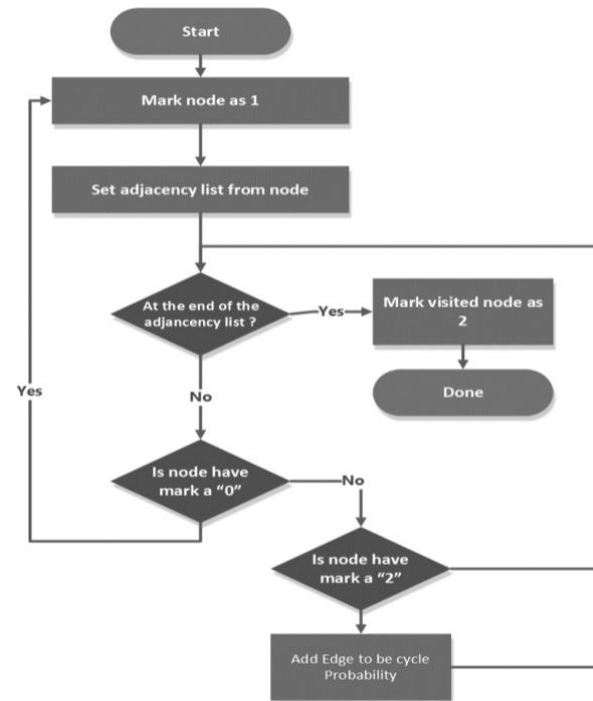


Figure 4: Flow chart detect back edge.

From the above Fig. 4., it can be seen that when back edge have been found, then origin node and current on going node will be saved.

2. Recognizing the cycle in this step can be performed with few steps on every graph part/model.

Those step can be listed as follow:

- a. DFS exploration
- b. Selection criteria

From previous process, back edge which already found indicates that those edge is a part of cycle path. Hence, to accurately found full cycle path, we can start to begin the search process by using DFS from saved back edges. This method can cut the searching time short without any needs to utilize DFS on every nodes available. See Fig. 5.

- a. The process continued with an exploration DFS in accordance with below flowchart with function "Recognizing Cycle". "Path" is considered a cycle if the first point is equal to the last point and the number of points in the path is at least 3.

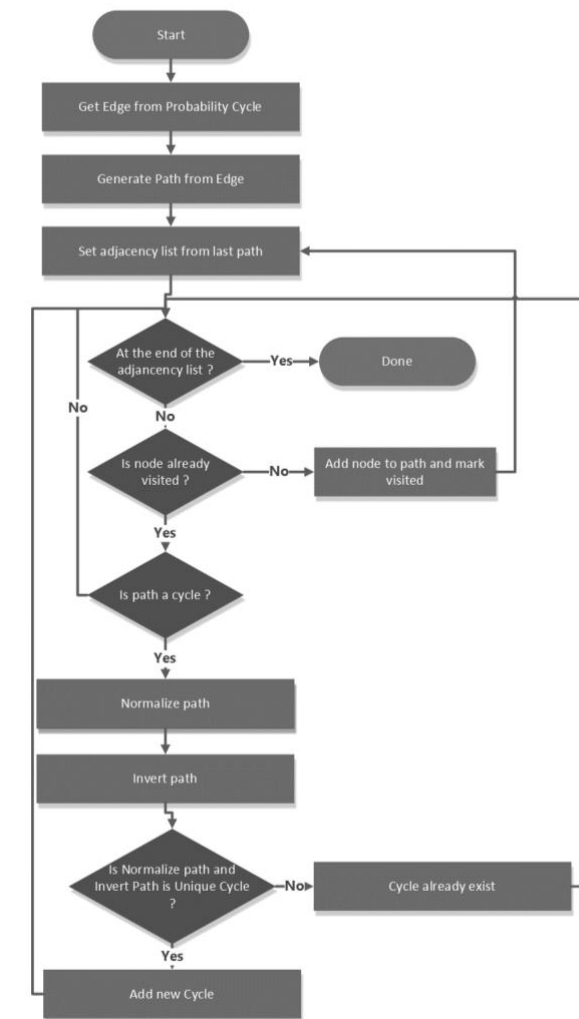


Figure 5: Flow chart recognize cycle.

- b. Selection criteria is an important stage which carried out to determine if the cycle is unique. Means cycle only can be found once. The method can be described as follow:

- 1) Normalize : One of the steps which is meant to create node sequence to form a cycle hence the number of nodes is sorted from the smallest, into the largest. See Fig. 6. From Fig. 6, we can acquire the node sequence which formed a cycle such 3,4,5,1,2. Where cycle reorganization can happen at random, in order to prevent any similar effort for the same cycle and to simplify the uniqueness from a cycle, the above founded cycle sequence need to be normalized. The normalized process consist of special rotation process with

the certain consistent order where the smallest node number and other following number will be placed on the beginning (first group) and all number before smallest node number will placed after that first group. Detail flowchart see Fig. 7.

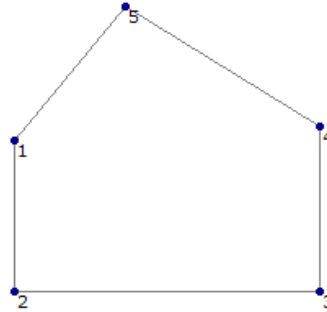


Figure 6: Example graph to explain normalize.

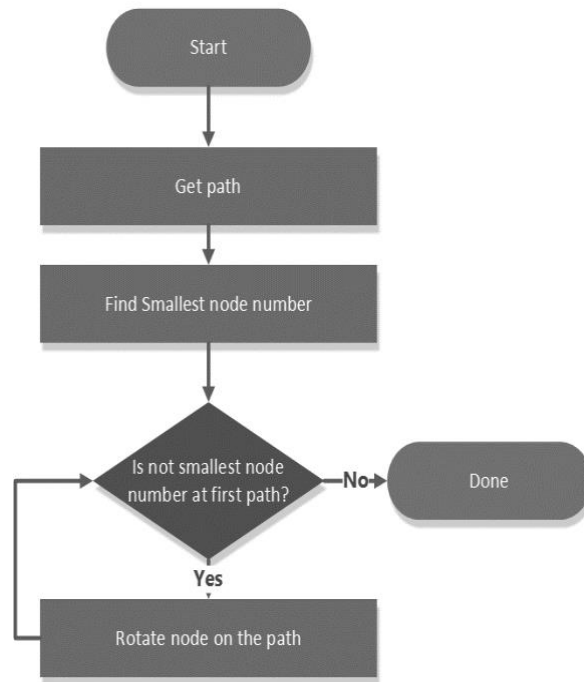


Figure 7: Flow chart normalized function.

- 2) Inverse sequence on this stage will be formed into new order of node sequence. which can be illustrated as follows:



Figure 8: Example graph to explain invers

From above Fig. 8. , the node sequence which can possibly formed is 7,8,9,6. After it going through normalization process, the output became 6,7,8,9 and inverse process will arrange the node sequence into 6,9,8,7. Although the resulted cycle (6,9,8,7) has a different node sequence from the normalized cycle(6,7,8,9), those two cycle originates from the same cycle. Thus the two cycles are considered equal. This flowchart below on “Fig. 9.” Describe detail step to create invers path:

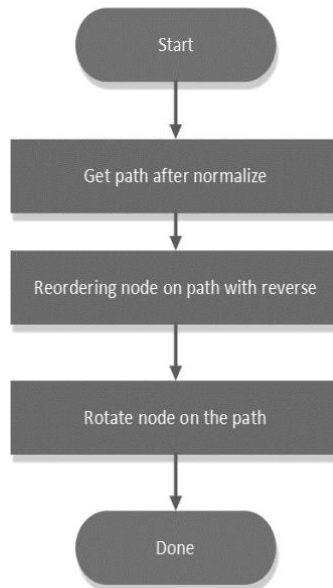


Figure 9: Flow chart invert function.

After the above procedure, the acquired cycle sequence including the invers result will be compared with the latest founded cycle. When it had been found that those cycles are unique (nothing can compare) then the cycle would be saved. Result: after all steps mentioned above have been carried out, and all possible cycle have been explored then in the next process, the cycles will be stored. In this stage, all cycles which have been found is unique and non-identical. And all recognized cycles are accurately possible to be fit on the related graph. However, there is still an overlapped cycle detected. This is undesirable because the scantling calculation only need unique cycle without any overlap, known as independent cycle.

3.5. Testing

Therefore algorithm procedure mentioned above have been tested on the simple model, as described on below on Fig. 10:

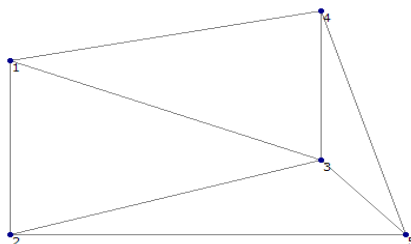


Figure 10: Simple graph containing cycle.

The calculation result using that algorithm mentioned above will recognized 13 possible cycle lists as follow:

- | | |
|--------------|--------------|
| 1) 1,2,3,5,4 | 8) 1,2,5,4 |
| 2) 2,3,5 | 9) 1,3,2,5,4 |
| 3) 1,3,5,2 | 10) 2,5,4,3 |
| 4) 1,3,5,4 | 11) 1,3,2 |
| 5) 1,4,3,5,2 | 12) 1,3,4 |
| 6) 3,5,4 | 13) 1,4,3,2. |
| 7) 1,2,5,4,3 | |

Those cycle lists are all the possible probability for any cycles that can be formed inside the graphic. Not all cycle on the list will be used on scantling calculation. The cycle that can be used on scantling calculation need to be defined as independent cycle.

- 1) 2,3,5
- 2) 3,5,4
- 3) 1,3,2
- 4) 1,3,4

3.6.Application Procedure on Ship Cross Section

In order to increase the accuracy of above developed procedure, then those procedure need to be applied on several cross section. The result from procedure application can be shown on Fig. 11- 13 below [7].

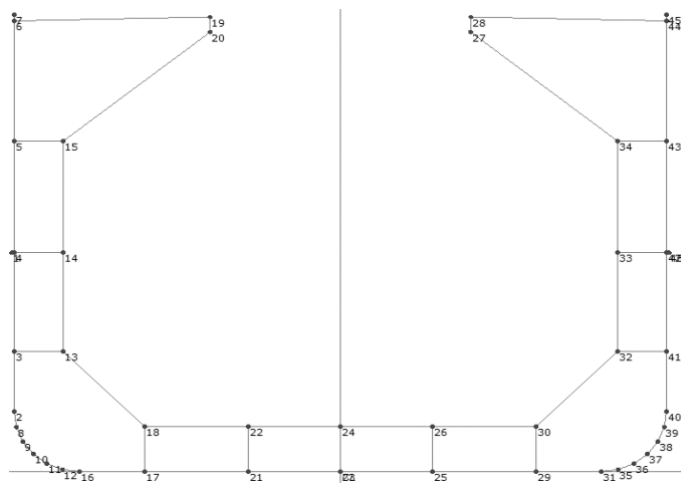


Figure 11: Model of bulk carrier.

Table 1: Result of bulk carrier model.

Node number	46
All cycle	78
Independent Cycle	12

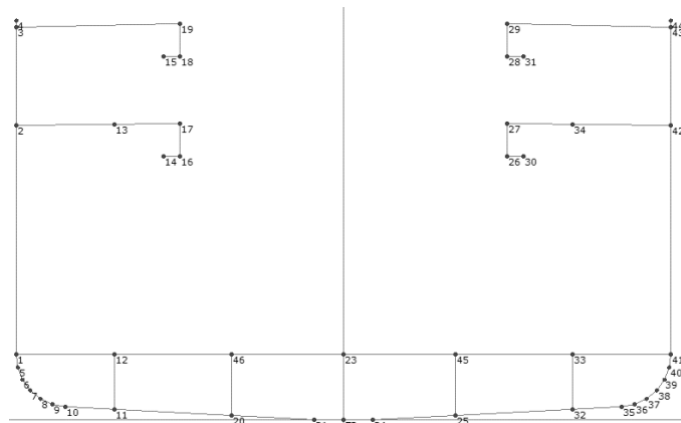


Figure 12: Model of general cargo.

Table 2: Result of general cargo model.

Node number	54
All cycle	21
Independent Cycle	6

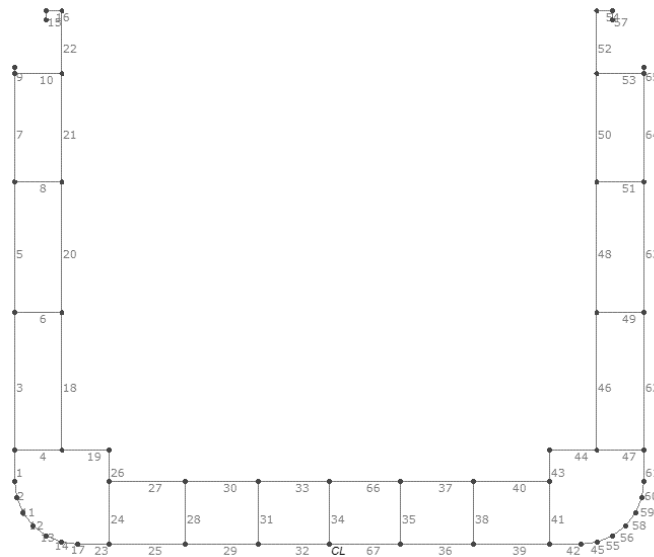


Figure 13: Model of container.

Table 3: Result of container model.

Node number	54
All cycle	105
Independent Cycle	14

From the Fig. 11-13 we can determine and decide that all the possible cycle have been found.

4. Conclusions

Research for algorithm development in order to recognize certain geometrical shape for scantling calculation has reach conclusion as follow:

- The developed procedure is capable to find all closed loop cycle from the undirected graph accurately.
- The process needed in finding cycle might became faster since not all the available nodes have been used as starting searching point. Hence the combination acquired is much smaller.
- Found Cycle list are considered unique, however overlapping cycle still occurred. therefore the further extraction process are need to be performed. the extraction process is too complex, to separated overlapcycle SAT theorem (separate axis theorem) may be used but that theorem has limitation, it only separates the convex shapes.

However, it is possible for cycle list to include concave shape. Therefore, the geometry of cycle which have concave shape needs to be split into convex shapes. This obstacle will be addressed in future work.

References

- [1] Makmun, S., Siswanto, dan Firmandha, T., "Numerical Evaluation of Normal Stress Distribution On Ship Cross". ICSOT Indonesia, 4-5 November 2015, Surabaya-Indonesia.
- [2] Firmandha, T., Makmun, S., dan Siswanto, "Numerical Evaluation of Shear Stress Distribution On Ship Cross". ICSOT Indonesia, 4-5 November 2015, Surabaya-Indonesia.
- [3] BKI, BKI Rules for Hull, Volume II (Pt. 1, Vol. II), 2017 Edition, BKI, Jakarta.
- [4] Donald Johnson. Finding all the elementary circuits of a directed graph. *SIAM J. Comput.*, 4(1):77–84, 1975.
- [5] R. Ferreira, R. Grossi, A. Marino, N. Pisanti, R. Rizzi, G. Sacomoto, *Optimal Listing of Cycles and st-Paths in Undirected Graphs*, 2012.
- [6] T.H.Cormen, C.E.Leiserson, R.L.Rivest, C.Stein. *Introduction of Algoritm 3rd edition*, 2009.
- [7] BKI.Dewaruci – Palapa II Software. 2017